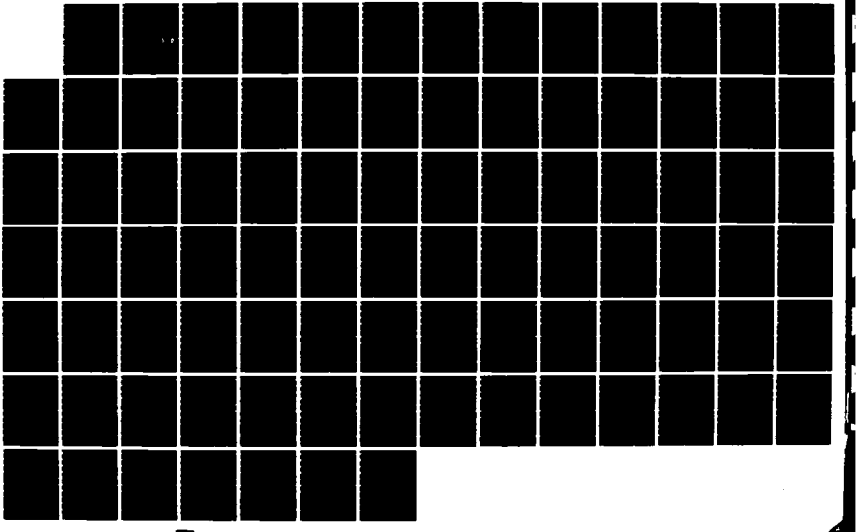
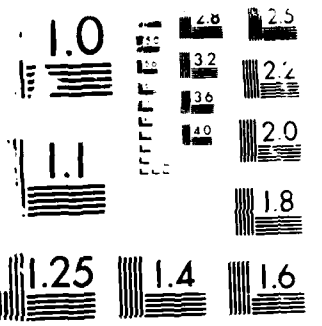
AD-A179 173 SOLUTION TO A CLASS OF ECONOMIC RELIABILITY PROBLEMS WITH GEOMETRIC PROGRAMMING(U) ARMY MILITARY PERSONNEL CENTER ALEXANDRIA VA C K OATNEY 14 APR 87 1/1

UNCLASSIFIED F/G 9/3 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

UNCLASSIFIED

DTIC FILE COPY

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1 REPORT NUMBER / 2 GOVT ACCESSION NO.	3 RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Solution to a Class of Economic Reliability Problems with Geometric Programming	5 TYPE OF REPORT & PERIOD COVERED Final Report 14 April 1987
	6 PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) CPT Cecilia K. Oatney	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Student HQDA, MILPERCEN ATTN: DAPC-OPA-E 200 Stovall Street, Alexandria, VA 22332	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS HQDA, MILPERCEN ATTN: DAPC-OPA-E 200 Stovall Street, Alexandria, VA 22332	12. REPORT DATE 14 April 1987
	13. NUMBER OF PAGES 85
14 MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited.

DTIC ELECTE APR 2 1 1987 S D

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

A thesis submitted to the Colorado School of Mines, Golden, CO 80401 in partial fulfillment of the requirements for the degree of Master of Science.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Reliability (to include cost), Geometric Programming, Electronic Equipment

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The rapid advancement of technology in the electronic industry has created a need for solving a certain class of large nonlinear reliability problems. The Department of Defense and civilian industries when planning for the allocations of budget resources require an efficient method for solving these problems. In certain areas a specific reliability, such as with nuclear reactors, must be attainable to ensure the accomplishment of the mission or to provide necessary safety factors. The current methods used to solve these

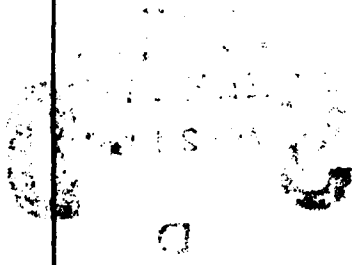
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A179 173

problems are very difficult and time consuming. A computer algorithm which is based on geometric programming has been developed to provide a quick, accurate, and efficient way of solving reliability problems.



Solution to a Class of Economic Reliability Problems with Geometric Programming

Cecilia K. Oatney, O-3
HQDA, MILPERCEN (DAPC-OPA-E)
200 Stovall Street
Alexandria, VA 22332

Final Report 14 April 1987

Approved for public release; distribution is unlimited.

A thesis submitted to the Colorado School of Mines, Golden, CO 80401
in partial fulfillment of the requirements for the degree of Master of Science.

T-3357

SOLUTION TO A CLASS OF ECONOMIC RELIABILITY PROBLEMS WITH GEOMETRIC PROGRAMMING

Accesion For	
NTIS CRA&I	✓
DTIC TAB	[ ]
Unannounced	[ ]
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

by

Cecilia K. Oatney

T-3357

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Master of Science (Mineral Economics).

Golden, Colorado
Date 9 April 1987

Signed: Cecilia K. Datney
Cecilia K. Datney

Approved: Ruth A. Maurer
Dr. Ruth A. Maurer
Thesis Advisor

Golden, Colorado
Date 9 April 1987

Dr. John A. Cordes
Associate Professor and Head
Mineral Economics Department

T-3357

ABSTRACT

The rapid advancement of technology in the electronic industry has created a need for solving a certain class of large nonlinear reliability problems. The Department of Defense and civilian industries when planning for the allocations of budget resources require an efficient method for solving these problems. In certain areas a specific reliability, such as with nuclear reactors, must be attainable to ensure the accomplishment of the mission or to provide necessary safety factors. The current methods used to solve these problems are very difficult and time consuming. A computer algorithm which is based on geometric programming has been developed to provide a quick, accurate, and efficient way of solving reliability problems.

TABLE OF CONTENTS

	Page
ABSTRACT	iii
LIST OF FIGURES AND TABLES	vi
ACKNOWLEDGMENTS	vii
DEDICATION	ix
CHAPTER	
1 INTRODUCTION	1
Problem Statement	1
The General Geometric Programming Approach	5
2 CURRENT METHODS OF SOLVING RELIBILITY PROBLEMS	8
3 METHOD FOR SOLVING MINIMUM COST PROBLEMS	12
Phase 1: The Objective Function	12
Phase 2: The Constraints	14
Phase 3: Solving by GP	18
Phase 4: Integer Solution	28
4 METHOD FOR SOLVING MAXIMUM COST PROBLEMS	32
Phase 1: Standard GP Form	32
Phase 2: Solving by GP	34
Phase 3: Integer Solution	40

5 SOLUTIONS TO APPLIED SAMPLE RELIABILITY PROBLEMS 45

Minimization Problems 46

Maximization Problems 49

6 CONCLUSIONS 54

REFERENCES 57

APPENDIXES

A FOUR RULES FOR GEOMETRIC PROGRAMMING 58

B COMPUTER ALGORITHM FOR SOLVING RELIABILITY PROBLEMS WITH GP USING MICROSOFT QUICKBASIC . 60

C SAMPLE COMPUTER RUN 75

FIGURES AND TABLES

Figure Page

1.1 General Reliability System 3

3.1 Brent-Dekker Method 25

Table

3.1 Minimum Integer Combinations 31

4.1 Maximum Integer Combinations 44

5.1 Minimum Two-Variable Solutions 47

5.2 Minimum Three-Variable Solutions 48

5.3 Minimum Four-Variable Solutions 49

5.4 Maximum Two-Variable Solutions 50

5.5 Maximum Three-Variable Solutions 51

5.6 Maximum Four-Variable Solutions 53

T-3357

ACKNOWLEDGMENTS

It is with sincere appreciation that I acknowledge the guidance and assistance of Dr. Ruth Maurer during the time I spent at the Colorado School of Mines. I could never have accomplished so much in so little time without her expertise.

I wish to acknowledge the significant assistance of Dr. Robert E. D. Woolsey in the development and preparation of this thesis. His knowlege and guidance, along with the atmosphere he fosters in the "bullpen" greatly enhanced my studies.

I wish to thank Dr. Robert Underwood and Lt. Col. Charles Nichols for their assistance and guidance on the final developments of my thesis. Their knowledge and expertise in diverse areas helped to guide me through various pitfalls.

I thank Lt. Col. James Thome for his constant assistance and encouragement throughout this thesis. I greatly appreciate his ability in helping me to keep the right perspective concerning my studies. I also thank his wife and sons for providing a family atmosphere on several occasions, which helped me keep my sanity.

I thank my fellow Army officers for providing an effective support element. I also wish to thank the ROTC cadre and cadets for including me in their activities and thus providing a means of keeping in contact with the Army.

I thank Gys Wessels, Norm Goddard, and Jean Goldberg for the various assistance they have provided. I thank Steven Strauss for his valuable assistance in developing the computer algorithm for this thesis.

Finally, I thank my family who have supported me in fulfilling my desire of getting my degree. Their encouragement has always been present, even if they couldn't be.

To

Mother and Father

Thank you for giving me the encouragement

to always strive for improvement.

And for teaching me not to be

afraid to work towards that

which is not easily

attainable.

Chapter 1

INTRODUCTION

Problem Statement

In the general field of electronic instrumentation, an important goal for both government and private industry is that of fielding systems that can operate reliably for a specified period of time within specified environments. One of the most difficult problems in this field is ensuring a level of acceptable reliability at minimum cost. Related to this problem is its economic dual, which is to maximize reliability within fixed cost limits. The purpose of this thesis is to show that these types of reliability problems are easily solvable, within certain restrictions, using geometric programming (GP).

Component reliability is defined as the mathematical probability that an individual component will continue performing its intended mission for a specific length of time. The failure rate of a particular component is stated as a probability between 0 and 1. In mathematical terms, component reliability equals 1 minus probability of failure, or 1 - P(failure). If a system is composed of more than one component in series, then the system reliability would be the product of the individual

component reliabilities. For one type of component, the system reliability would become

$1 - P(\text{failure})^{\#\ (\text{of units})}$.

Although the definition of component reliability is straightforward, systems involving reliability can become rather complex. Figure 1.1 depicts a general reliability system. The system consists of n elements where each element can have one or more redundant or parallel components.

The reliability of any separate element and the reliability of the system itself are increased exponentially through redundancy of components. However, one must then face the corresponding multiplicative increase in cost.

The first problem of achieving acceptable reliability at minimum cost could be stated as follows. Consider a system involving electronics with n interconnected elements. Each element is made up of some number of backup components that will switch on in case of failure. Figure 1.1 is such a system. A two-element system, where we wish to minimize the cost of parallel back-up components subject to a minimally acceptable level of total system reliability, would be formulated as

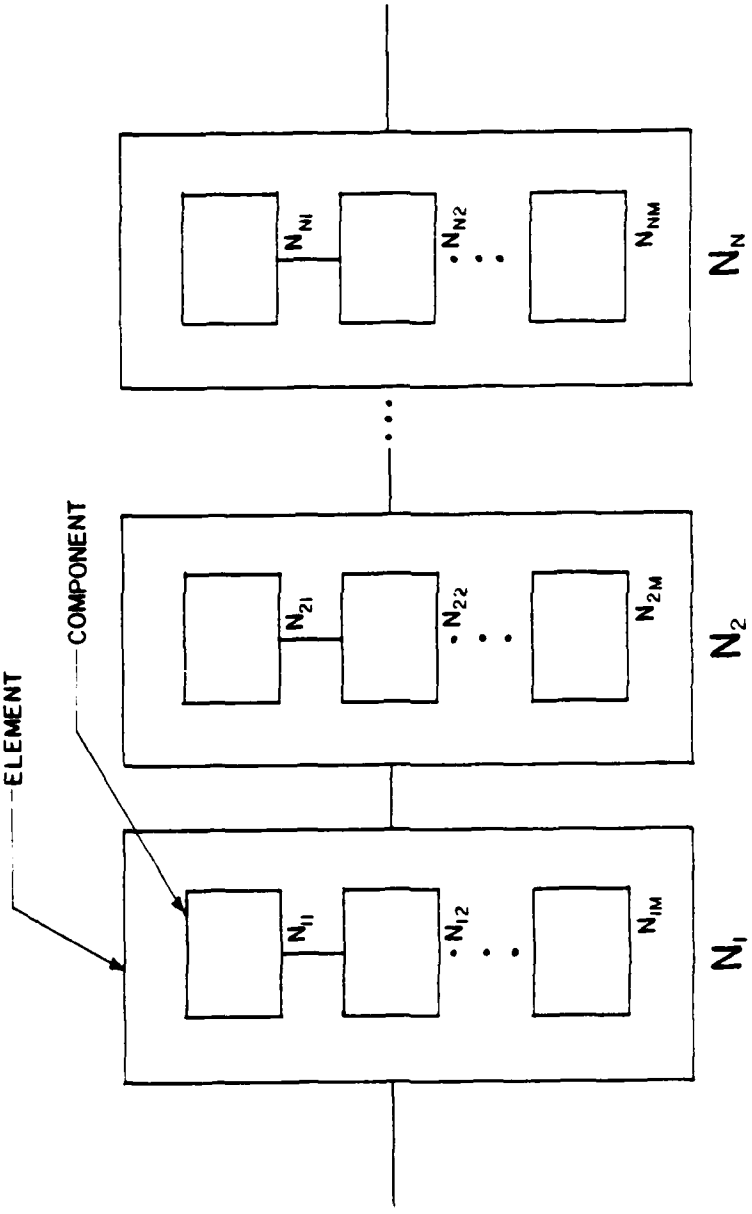


Figure 1.1 Diagram of a General Reliability System

$$\text{minimize cost:} \quad C = \$5N_1 + \$7N_2$$

$$\text{subject to:} \quad \left(1 - 0.4^{N_1}\right)\left(1 - 0.6^{N_2}\right) \geq 0.90 \quad (1.1)$$

where N_1 represents the number of components of element 1 and N_2 represents the number of components of element 2. The cost of each N_1 component is \$5; for each N_2 component it is \$7. The probability of failure of each component of element 1 is 0.4; the probability of failure of each component of element 2 is 0.6; while the minimally acceptable level of total system reliability is 0.90.

The second problem, the economic dual, also considers a system involving electronics with n interconnected elements. As before, each element is made up of a number of backup components that will switch on in case of failure. However, in the case of the two-element system shown below, the problem is maximization of total reliability given a total cost constraint, e.g., \$60.

$$\text{maximize reliability:} \quad R = \left(1 - 0.4^{N_1}\right)\left(1 - 0.6^{N_2}\right)$$

$$\text{subject to:} \quad \$5N_1 + \$7N_2 \leq \$60 \quad (1.2)$$

This thesis will develop an algorithm for the solution of both types of problems. The algorithm, using geometric

programming, will be programmed for interactive use in Basic, and a class of appropriate test problems will be used to demonstrate the algorithm's ability to achieve global optimality.

The General Geometric Programming Approach

Problems involving reliability are normally characterized as nonlinear design problems. The objective function and/or the constraint functions take the generalized nonlinear form of

$$\text{minimize } g(X) = \sum_{i=1}^{n} C_i P_i(X)$$

where

$P_i(X) = X_1^{a_{i1}} X_2^{a_{i2}} \ldots X_n^{a_{in}}$, $i = 1, 2, \ldots, n$;

C_i and a_{ij} are typically physical constants; and

X_j are design variables, $j = 1, 2, \ldots, n$.

Geometric programming (GP) is a mathematical programming technique used to solve such nonlinear design problems. The general form of a GP problem is

$$\text{minimize cost: } C = g_0(X)$$

subject to: $g_i(X) \leq 1 \qquad i = 1, 2, \ldots, m$

$X_j > 0 \qquad j = 1, 2, \ldots, n$

where

$$g_i(X) = \sum_{t=1}^{T_i} C_{it} \prod_{j=1}^{n} X_j^{a_{ijt}}$$

A few definitions used in GP are helpful in order to further understand and discuss the algorithm. A "term" is defined as a grouping of one or more variables with or without coefficients separated from any other grouping of variables by a plus, minus, or inequality sign. The degree of difficulty is an indication of how difficult it is to solve the original problem; progressively higher numbered problems become more difficult to solve. The degree of difficulty (DD) of a particular problem is determined by taking the number of individual terms in the entire problem, minus the number of variables minus 1. Thus, a GP problem such as

minimize cost: $C = \underbrace{\$12N_1 9N_2}_{\text{term 1}}$

subject to: $\underbrace{N_1^{-1}}_{\text{term 2}} \geq 0.95$

$$\underbrace{N_2^{-1}}_{\text{term 3}} \geq 0.95$$

would have a DD of zero, as 3 terms minus 2 variables (N_1 and N_2) minus 1 equals zero.

In GP, a zero DD problem is desirable, as a higher DD problem would require some type of mathematical manipulation before the problem could be solved. The reliability problems solved using the algorithm developed below will be one DD, and therefore will require an internal mathematical method to solve for the unknown δ values. This method will be addressed in Chapter 3. The first step in developing the required algorithm is to rewrite the problem using various substitutions to manipulate the variables so that the resulting GP problem is one DD.

Chapter 2

CURRENT METHODS OF SOLVING RELIABILITY PROBLEMS

Reliability as a separate mathematical science was developed as a direct result of analyses conducted during World War II. The rapid automation of technology created a need for a way to determine the reliability of specific items of equipment as well as total systems.

At the close of World War II, the military conducted several studies on reliability. These studies revealed some startling facts: (1) electronic equipment used during Navy maneuvers was operational only 30% of the time; (2) 67-75% of the Army electronic equipment was out of commission or under repairs; (3) Air Force repair and maintenance costs were ten times the original equipment costs; (4) for every electonic tube on the shelf, seven were in transit; (5) approximately one electronic technician was needed for every 250 electronic tubes; and (6) the number of electronic tubes required on board a destroyer had risen from 60 to 3,200 (Shooman, 1968).

In 1950, the Department of Defense established an ad hoc committee on reliability, which was replaced in 1952 by the Advisory Group on the Reliability of Electronic

Equipment (AGREE). AGREE and the Institute of Electrical and Electronic Engineers (IEEE) are currently the main agencies conducting reliability analysis.

Today reliability is stressed in military applications of electronics, aviation, and weapons systems. Reliability has become an important aspect of military contracting as it is essential in evaluating the system usefulness or goodness along with cost, size, and weight.

The approach to solving reliability problems varies with each individual and organization. Most often, reliability problems are treated as applied probability or statistics problems. Because analysis of complicated reliability problems becomes lengthy or difficult, transform methods or computer solutions are necessary. Complex systems are almost impossible to solve in entirety and must be decomposed into functional areas.

Currently, reliability problems of the type described in Chapter 1 are theoretically solved using the Kuhn-Tucker conditions. In reality, most reliability problems are solved using either a variable search or Markov process method. It is fairly simple to solve reliability problems using the Markov process; however, the labor involved increases tremendously as the number of elements in the

total system expands. In addition, this method cannot guarantee global optimality; geometric programming, on the other hand, does guarantee global optimality.

GP is practical for solving reliability problems since it is capable of exploiting the linear algebraic structure of each problem. These linearities can appear as linear equations, linear inequalities, or as matrices associated with nonlinearities.

Research on the type of reliability problems discussed in Chapter 1 revealed only one previous attempt to use GP. In 1968, A. J. Federowicz and M. Mazumdar, while working for the Westinghouse Research Laboratories, developed a method of using substitution to enable the given problem to be written in standard GP form. Once the problem was written in GP form it was easier to solve.

The algorithm developed in this thesis follows the initial substitution and manipulation that Federowicz and Mazumdar used to solve maximization problems. Instead of using Federowicz and Mazumdar's method, which uses logarithms and differentiation to find the unknown t_i values, either the quadratic equation or the Brent-Dekker method was used here to find the positive root of the unkown t_i. A method for solving minimization problems has also been

included. Further comparison between Federowicz and Mazumdar's method and the one developed in this thesis will be addressed in Chapter 5.

Chapter 3
METHOD FOR SOLVING MINIMUM COST PROBLEMS

There are four phases in the GP method for solving minimum cost problems and in developing an associated generalized computer algorithm. Manipulation of terms and substitution in the early phases can reformulate the original problem in the standard GP form used by Woolsey's "Quick & Dirty" four rule algorithm (see Appendix A).

Phase 1: The Objective Function

This phase consists of three steps and uses substitution and the rules of logarithms to manipulate the objective function of the original problem. In the case of the first problem discussed, minimizing cost subject to a specified reliability, the generalized form of the objective function is

$$\text{minimize cost: } C = \sum_{j=1}^{n} C_j N_j$$

where

$$j = 1, 2, \ldots, n$$

To transform the objective function into the form required by GP, consider the following procedure.

Step 1) Using substitution, let

$$N_j = \ln X_j$$

Step 2) Rewrite the objective function as

minimize cost: $C = \sum_{j=1}^{n} C_j \ln X_j$

Step 3) Use the rules of logarithms to get the objective function in standard GP form.

a) When the antilog of a function such as

$$\ln Y_j = C_j \ln X_j$$

is taken, it becomes

$$Y_j = X_j^{C_j}$$

b) When the antilog of an additive function such as

$$\ln Y = C_1 \ln X_1 + C_2 \ln X_2$$

is taken, it becomes multiplicative

$$Y = X_1^{C_1} X_2^{C_2}$$

Transform the original objective function of sample problem (1.1) by performing the above steps.

Step 0) Restate the original objective function.

minimize cost: $C = \$5N_1 + \$7N_2$

Steps 1 & 2) Rewrite the above equation as discussed.

minimize cost: $C = 5 \ln X_1 + 7 \ln X_2$

Step 3) Take the antilog of the above equation.

minimize cost: $e^C = X_1^5 X_2^7$

The objective function is now in standard GP form.

Phase 2: The Constraints

This phase consists of *nine* steps and uses substitution and the rules of logarithms to manipulate the constraint of the original problem. The generalized form of the orignal constraint discussed is

$$\left(1 - P_1^{N_1}\right)\left(1 - P_2^{N_2}\right) \cdots \left(1 - P_n^{N_n}\right) \geq b$$

Step 1) The reliability of the jth element stated as

$$1 - P_j^{N_j}$$

is difficult to handle in GP. Use substitution to get rid of this "messy" term by letting

$$Z_j = 1 - P_j^{N_j}$$

Step 2) Rewrite the constraint as

$$Z_1 Z_2 \ldots Z_n \geq b$$

Step 3) At this point, because of the substitution previously performed, the objective function is written in terms of X_j, while the constraint is written in terms of Z_j. The problem cannot be solved in this form, as there is now no relation between the variables in the objective function and the constraint. In GP, all constraints will be tight at optimality (Duffin et al., 1967). Therefore, an additional constraint (written as an inequality) can be added for each Z_j, showing the substitution used in Step 1.

$$Z_j \leq 1 - P_j^{N_j}$$

Step 4) Terms that have a constant raised to a variable power are rather difficult to handle with GP. Thus, let

$$Y_j = P_j^{N_j}$$

Step 5) Take the natural log of both sides

$$\ln Y_j = N_j \ln P_j$$

Step 6) Use the substitution $N_j = \ln X_j$, as was done for the objective function, to obtain

$$\ln Y_j = \ln X_j \ln P_j$$

or

$$\ln Y_j = \ln P_j \ln X_j$$

Step 7) Take the antilog of the above equation.

$$Y_j = X_j^{\ln P_j}$$

Step 8) Rewrite each added constraint (as required in Step 3), using this manipulation to obtain

$$Z_j \leq 1 - X_j^{\ln P_j}$$

Step 9) Manipulate each constraint, so that the right-hand sides are less than or equal to 1.

a) Dividing both sides by Z_j, the first constraint becomes

$$1 \geq b\left(Z_1^{-1} Z_2^{-1} \ldots Z_n^{-1}\right)$$

b) Rewritten, it becomes

$$b\left(Z_1^{-1}Z_2^{-1}\ldots Z_n^{-1}\right) \leq 1$$

c) The added constraints become

$$Z_j + X_j^{\ln P_j} \leq 1$$

By performing the above nine steps to the constraint of the original sample problem (1.1), the resulting constraints are transformed into standard GP form. An application of the above nine steps is shown below.

Step 0) Restate the original constraint:

$$\left(1 - 0.4^{N_1}\right)\left(1 - 0.6^{N_2}\right) \geq 0.90$$

Steps 1 & 2) Replace the "messy" terms with Z_j, and rewrite the constraint:

$$Z_1 Z_2 \geq 0.90$$

Steps 3-8) Add the additional contraints for each Z_j as discussed:

$$Z_1 \geq 1 - X_1^{\ln 0.4}$$

$$Z_2 \geq 1 - X_2^{\ln 0.6}$$

Step 9) Rewrite each of the constraints as discussed.

$$0.9Z_1^{-1}Z_2^{-1} \leq 1$$

$$Z_1 + X_1^{\ln 0.4} \leq 1$$

$$Z_2 + X_2^{\ln 0.6} \leq 1$$

The constraints are now written in standard GP form.

Phase 3: Solving by GP

This phase consists of seven steps, and uses Woolsey's four rules to solve the problem. Phases 1 and 2 have reformulated sample problem (1.1) as follows.

minimize cost: $C = \underbrace{X_1^5 X_2^7}_{\text{term 1}}$

subject to: $\underbrace{0.90Z_1^{-1}Z_2^{-1}}_{\text{term 2}} \leq 1$

$$\underbrace{Z_1}_{\text{term 3}} + \underbrace{X_1^{\ln 0.4}}_{\text{term 4}} \leq 1$$

$$\underbrace{Z_2}_{\text{term 5}} + \underbrace{X_2^{\ln 0.6}}_{\text{term 6}} \leq 1$$

At this point the resulting GP problem is one DD as there are 6 terms and 4 variables (X_1, X_2, Z_1, and Z_2). Using Woolsey's four rules of geometric programming (Appendix A), we can solve sample problem (1.1).

Step 1) Write the form of the optimal solution according to Rule 1.

$$\text{minimize cost: } C = (1/\delta_1)^{\delta_1}(0.9/\delta_2)^{\delta_2}(\delta_2)^{\delta_2}(1/\delta_3)^{\delta_3}(1/\delta_4)^{\delta_4}$$

$$(\delta_3 + \delta_4)^{(\delta_3 + \delta_4)}(1/\delta_5)^{\delta_5}(1/\delta_6)^{\delta_6}(\delta_5 + \delta_6)^{(\delta_5 + \delta_6)}$$

Step 2) State the exponent matrix of the δ's according to Rule 2.

$$\begin{aligned}
&(OF): && \delta_1 && && && = 1 && (3.1)\\
&(X_1): && 5\delta_1 && + \ln 0.4\delta_4 && && = 0 && (3.2)\\
&(X_2): && 7\delta_1 && && + \ln 0.6\delta_6 && = 0 && (3.3)\\
&(Z_1): && -\delta_2 + \delta_3 && && && = 0 && (3.4)\\
&(Z_2): && -\delta_2 && && + \delta_5 && = 0 && (3.5)
\end{aligned}$$

Step 3) Solve for the values of the δ's using the logic that follows. A pattern occurs in the δ matrix table. Regardless of the number of variables in the original problem, this pattern will always be present. The pattern is shown below.

$\delta_1 = 1$

For j = 1 to the total number of variables

$$\delta_2 = \delta_{(2*j + 1)} = \text{UNKNOWN}$$

and

$$\delta_{(2*j + 2)} = -\text{Cost}_j \; / \; \ln \text{Failure}_j$$

The following steps depict how the pattern works for the two-variable case.

a) From equation (3.1) determine that

$$\delta_1 = 1$$

b) From equation (3.2) determine that

$$\delta_4 = 5.4567833$$

c) From equation (3.3) determine that

$$\delta_6 = 13.7033060$$

d) From equation (3.4) determine that

$$\delta_2 = \delta_3$$

e) From equation (3.5) determine that

$$\delta_2 = \delta_5$$

f) Thus

$$\delta_2 = \delta_3 = \delta_5$$

Step 4) Skip to Rule 4 since the values of δ_2, δ_3, and δ_5 are unknown.

a) Write the equations for δ_3 and δ_5.

$$\delta_3 = Z_1(\delta_3 + \delta_4)$$

$$\delta_5 = Z_2(\delta_5 + \delta_6)$$

b) Solve each equation in terms of Z_j.

$$Z_1 = \frac{(\delta_3)}{(\delta_3 + \delta_4)}$$

$$Z_2 = \frac{(\delta_5)}{(\delta_5 + \delta_6)}$$

c) Replace the δ's with the known values.

$$Z_1 = \frac{(\delta_2)}{(\delta_2 + 5.4567833)}$$

$$Z_2 = \frac{(\delta_2)}{(\delta_2 + 13.703306)}$$

d) At optimality each constraint is tight, thus make the original constraint an equality and solve it in terms of Z_1Z_2.

$$Z_1Z_2 = 0.90$$

e) Substitute the equations found in Step 4c for Z_1Z_2 in the above equation:

$$\frac{(\delta_2)}{(\delta_2 + 5.4567833)} \times \frac{(\delta_2)}{(\delta_2 + 13.703306)} = 0.90$$

Thus,

$$.1\delta_2^2 - 17.244085_2 - 67.298374 = 0$$

Step 5) Solve the above equation. The power of the first term determines the method to be used.

a) Since the power of the first term is 2 and the equation in Step 5e is in the quadratic form,

$$a\delta_2^2 \pm b\delta_2 \pm c = 0$$

solve for δ_2 using the quadratic formula.

$$\delta_2 = \frac{-b \pm \left(b^2 - 4ac\right)^{\frac{1}{2}}}{2a}$$

b) Substitute in the values found in Step 4e and solve for λ_2:

$$\lambda_2 = \frac{-(-17.24408) \pm \left[(17.24408)^2 - 4(.1)(-67.29837)\right]^{\nu}}{2(.1)}$$

Thus,

$$\lambda_2 = 176.25895$$

and

$$\lambda_2 = -3.818155$$

c) Since in GP the λ's must be nonnegative, only the real positive root of the quadratic equation can be used; therefore, drop the negative λ_2.

$$\lambda_2 = \lambda_3 = \lambda_5 = 176.25895$$

d) If the power of the first term is greater than 2, then to solve for the unknown λ_j a modified secant method may be used. This method, known as the Brent-Dekker method, is based upon both bisection and the secant rule methods. It is similiar to the Newton-Raphson method, except only the ability to evaluate f(x) is necessary. The Newton-Raphson method is slightly faster, but it requires the first

approximation concerning the bracketing of the roots to be close. In addition, the Newton-Raphson method may not converge, whereas the Brent-Dekker method is guaranteed to work once the function is bracketed.

To start the Brent-Dekker method, attempt to bracket the function with two values (B and C). Figure 3.1 depicts this starting bracket. The function must be continuous, and f(B)f(C) must be less than zero. Throughout this algorithm, B is assumed to be the better root.

Convergence is based on a mixed relative-absolute error test where

$$\left|\frac{C - B}{2}\right| \leq \max[\text{Absolute Error}, |B| * \text{Relative Error}]$$

The secant rule calculates the next iterate (D) by starting with the first two iterates A and B. The variable A is initially set equal to C. Thus,

$$D = B - f(B)\frac{B - A}{f(B) - f(A)}$$

Figure 3.1 depicts how the starting bracket is used to find the next iterate D using this

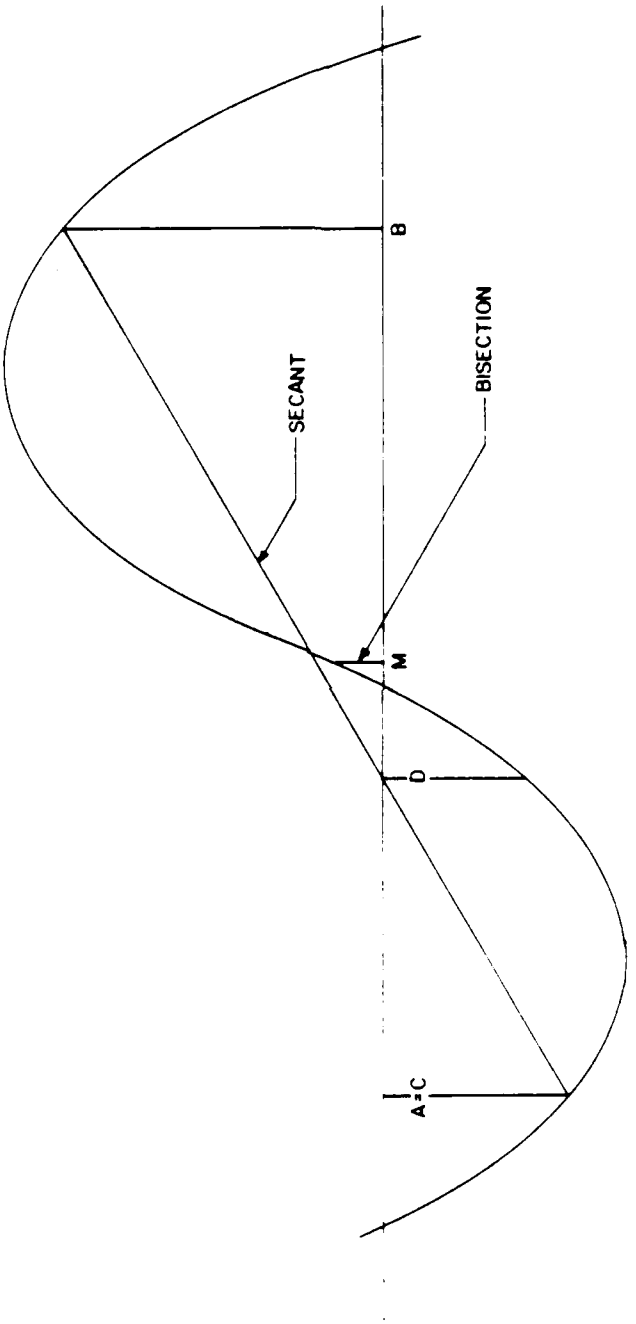


Figure 3.1 Brent-Dekker Method of Finding Roots

method. If D is to become the next iterate, then it must lie between B and the midpoint between B and C. If D is between B and the midpoint, then it is set equal to B. The old B is then set equal to C. If D is not in the interval between B and the midpoint, then the midpoint between B and C is used as the new iterate.

The Brent-Dekker method guarantees that there is either a root of $f(x) = 0$ in the interval between B and C, or that one of the end points is as close to a root as the desired precision permits, if the function being evaluated is continuous and the original bracket (B,C) results in $f(B)f(C) < 0$.

Step 6) Solve for X_1 and X_2 according to Rule 4, by replacing the δ's with their known values.

a) Write the equations for δ_4 and δ_6.

$$\delta_4 = X_1^{\ln 0.4}(\delta_3 + \delta_4)$$

$$\delta_6 = X_2^{\ln 0.6}(\delta_5 + \delta_6)$$

b) Substitute in the known values for the δ's.

$$5.4567833 = X_1^{-0.9162907}(176.25095 + 5.4567833)$$

$$13.703306 = X_2^{-0.5108256}(176.25895 + 13.703306)$$

c) Solve the above equations for X_1 and X_2.

$$X_1 = 45.871487$$

$$X_2 = 171.904510$$

Step 7) Use the above values to solve for the objective function.

a) Find the optimal GP values of N_1 and N_2 by substituting in the values for the ln X_j.

$$N_1 = \ln X_1 = \ln 45.871487 = 3.8258437$$

$$N_2 = \ln X_2 = \ln 171.90451 = 5.1469392$$

b) Therefore, the optimal GP value of the objective function is

minimum cost: C = \$55.16

Phase 4: Integer Solution

Components cannot be split into fractions, therefore an integer solution is required. This phase will consist of four steps that will ultimately provide the minimum cost integer component solution.

Step 1) Round each N_j down to the nearest integer, then add 1, and write its value.

$$N_1 = 4$$

$$N_2 = 6$$

Step 2) This value will more than satisfy the original constraint; however, it may not be the optimal combination of variables that both satisfies the constraint and has the minimum cost. Determine the reliability and cost.

$$\text{reliability} = \left(1 - 0.4^4\right)\left(1 - 0.6^6\right) = 0.9289384 > 0.90$$

$$\text{cost} = \$5(4) + \$7(6) = \$62.00$$

Step 3) Determine the number of combinations that must be compared to ensure that the optimal solution is found.

a) Add 1 to each integer value found in Step 1. Determine the new reliability for each element.

reliability element 1 = $1 - 0.4^5 = 0.989760$

reliability element 2 = $1 - 0.6^7 = 0.9720064$

b) Use the following formula developed by Steven A. Strauss to find the number of combinations to be checked to ensure that the minimum cost is found subject to the minimum required reliability.

$$\pm \text{ combination } N_j = \pm \text{ integer}(N_j) - \text{integer}$$

$$\text{of } \frac{[\ln(1 - \text{minimum required reliability}/k)]}{\ln [P(\text{failure of } j)]}$$

where

k = reliability of each element found in Step 3a, times all the other element reliabilities except for the jth element (the one whose optimal number of combinations is currently being calculated)

Therefore,

$\pm$ combination N_1 = $\pm$ 4 - integer

$$\text{of } \frac{[\ln (1 - 0.90/0.9720064)]}{\ln 0.40} = \pm 2$$

$\pm$ combination N_2 = $\pm$ 6 - integer

$$\text{of } \frac{[\ln (1 - 0.90/0.9897600)]}{\ln 0.60} = \pm 2$$

c) Find all combinations for each variable as determined in Step 3b. The combinations are also a sensitivity analysis for reliability and cost. Table 3.1 shows the combinations of N_1 and N_2 which result for example problem (1.1).

Step 4) Select the feasible combination with the minimum cost. This will be the optimal integer solution.

minimum cost: $C^* = \$60.00$

$N_1^* = 5$

$N_2^* = 5$

Table 3.1 Minimum Integer Combinations

N_1	N_2	RELIABILITY	COST (IN DOLLARS)
2	4	0.7311360*	38.00
2	5	0.7746816*	45.00
2	6	0.8008090*	52.00
2	7	0.8164854*	59.00
2	8	0.8258912	66.00
3	4	0.8146944*	43.00
3	5	0.8632166*	50.00
3	6	0.8923300	57.00
3	7	0.9097980	64.00
3	8	0.9202788	71.00
4	4	0.8481178*	48.00
4	5	0.8986307	55.00
4	6	0.9289384	62.00
4	7	0.9471230	69.00
4	8	0.9580338	76.00
5	4	0.8614871*	53.00
5	5	0.9127963	60.00
5	6	0.9435818	67.00
5	7	0.9620531	74.00
5	8	0.9731358	81.00
6	4	0.8668348*	58.00
6	5	0.9184625	65.00
6	6	0.9494391	72.00
6	7	0.9680251	79.00
6	8	0.9791766	86.00

* denotes those combinations which are not feasible since they are less than the minimum required reliability.

Chapter 4

METHOD FOR SOLVING MAXIMUM RELIABILITY PROBLEMS

There are three phases in the method for solving maximum reliability problems and in developing an associated generalized computer algorithm. This chapter uses the same manipulation of terms and substitution for the objective function and constraints as does Chapter 3. The second problem, maximizing reliability subject to a specified cost limitation, would be written in generalized form as

$$\text{maximize reliability:} \quad R = \left(1 - P_1^{N_1}\right)\left(1 - P_2^{N_2}\right) \cdots \left(1 - P_n^{N_n}\right)$$

$$\text{subject to:} \quad \sum_{j=1}^{n} C_j N_j \leq b$$

where

$$j = 1, 2, \ldots, 4$$

Phase 1: Standard GP Form

This phase consists of three steps used to get the original sample problem (1.2) into standard GP form. This is accomplished by using the substitutions and manipulations described in Chapter 3. The substitutions include

$$N_i = \ln X_i$$

and

$$Z_j = 1 - X_j^{\ln P_j}$$

Step 1) Restate the original sample problem (1.2).

maximize reliability: $R = \left(1 - 0.4^{N_1}\right)\left(1 - 0.6^{N_2}\right)$

subject to: $\$5N_1 + \$7N_2 \leq \$60$

Step 2) Use the substitution and manipulation as described in Chapter 3 to rewrite sample problem (1.2).

maximize reliability: $R = Z_1 Z_2$

subject to: $x_1^5 x_2^7 \leq e^{60}$

$$Z_1 \leq 1 - x_1^{\ln 0.4}$$

$$Z_2 \leq 1 - x_2^{\ln 0.6}$$

Step 3) GP only handles minimization problems: therefore, to change the objective function to a minimization, invert the function. The resulting problem becomes

minimize reliability: $R^{-1} = \underbrace{Z_1^{-1}Z_2^{-1}}_{\text{term 1}}$

subject to: $\underbrace{e^{-60}X_1^5X_2^7}_{\text{term 2}} \leq 1$

$\underbrace{Z_1}_{\text{term 3}} + \underbrace{X_1^{\ln 0.4}}_{\text{term 4}} \leq 1$

$\underbrace{Z_2}_{\text{term 5}} + \underbrace{X_2^{\ln 0.6}}_{\text{term 6}} \leq 1$

Phase 2: Solving by GP

This phase consists of six steps and uses Woolsey's four rules to solve the problem. Sample problem (1.2) is now in standard GP form and is a one DD problem as there are 6 terms and 4 variables (X_1, X_2, Z_1, and Z_2). The following steps are the same as thosed used in Phase 3 of Chapter 3.

Step 1) Write the form of the optimal solution according to Rule 1:

minimize reliability: $R^{-1} = (1/\delta_1)^{\delta_1}(e^{-60}/\delta_2)^{\delta_2}(1/\delta_3)^{\delta_3}$
$(1/\delta_4)^{\delta_4}(\delta_3 + \delta_4)^{(\delta_3 + \delta_4)}(1/\delta_5)^{\delta_5}(1/\delta_6)^{\delta_6}(\delta_5 + \delta_6)^{(\delta_5 + \delta_6)}$

Step 2) State the exponent matrix of the δ's according to Rule 2.

$$(OF): \quad \delta_1 = 1 \quad (4.1)$$
$$(Z_1): \quad -\delta_1 + \delta_3 = 0 \quad (4.2)$$
$$(Z_2): \quad -\delta_1 + \delta_5 = 0 \quad (4.3)$$
$$(X_1): \quad + 5\delta_2 + \ln 0.4\delta_4 = 0 \quad (4.4)$$
$$(X_2): \quad + 7\delta_2 + \ln 0.6\delta_6 = 0 \quad (4.5)$$

Step 3) Solve for the values of the δ's using the logic that follows. A pattern occurs in the δ matrix. Regardless of the number of variables in the original problem, this pattern will always be present. The pattern is shown below.

$\delta_1 = 1$

For j = 1 to the number of variables

$\delta_{(2*j + 1)} = \delta_1 = 1$

$\delta_{(2*j + 2)} = (-\text{Cost}_j / \ln \text{Failure}_j)\delta_2$

The following steps depict how the pattern works for the two-variable case.

a) From equation (4.1) determine that

$$\delta_1 = 1$$

b) From equation (4.2) determine that

$$\delta_3 = 1$$

c) From equation (4.3) determine that

$$\delta_5 = 1$$

d) From equation (4.4) determine that

$$\delta_2 = 0.1832581\delta_4$$

e) From equation (4.5) determine that

$$\delta_2 = 0.0729751\delta_6$$

f) Solve each unknown δ in terms of δ_2, therefore

$$\delta_4 = 5.4567833\delta_2$$

and

$$\delta_6 = 13.703306\delta_2$$

Step 4) Skip to Rule 4 since the values of δ_2, δ_4 and δ_6 are unknown.

a) Write the equations for δ_4 and δ_6.

$$\delta_4 = X_1^{\ln 0.4}(\delta_3 + \delta_4)$$

$$\delta_6 = X_2^{\ln 0.6}(\delta_5 + \delta_6)$$

b) Solve each equation in terms of X_j.

$$X_1 = \left[\frac{(\delta_3 + \delta_4)}{(\delta_4)}\right]^{(1/0.9162907)}$$

$$X_2 = \left[\frac{(\delta_5 + \delta_6)}{(\delta_6)}\right]^{(1/0.5180256)}$$

c) Replace the δ's with known values.

$$X_1 = \left[\frac{(1 + 5.4567833\delta_2)}{(5.4567833\delta_2)}\right]^{1.0913567}$$

$$X_2 = \left[\frac{(1 + 13.703306\delta_2)}{(13.703306\delta_2)}\right]^{1.9576152}$$

d) At optimality each constraint is tight, thus make the original constraint an equality and solve it in terms of X_1X_2.

$$x_1^5x_2^7 = e^{60}$$

e) Substitute the equations found in Step 4c for X_1X_2 in the above equation, and raise each variable to the appropriate power.

$$e^{60} = \left[\frac{(1 + 5.4567833\delta_2)}{(5.4567833\delta_2)}\right]^{(1.0913567)(5)}$$

$$\text{times} \qquad \left[\frac{(1 + 13.703306\delta_2)}{(13.703306\delta_2)}\right]^{(1.9576152)(7)}$$

Step 5) The above equation is very messy, hence all maximization problems regardless of the number of variables will be solved using the Brent-Dekker method discussed in Chapter 3. With a starting bracket (B,C) of (0.001,1000) the value for δ_2 is found to be

$$\delta_2 = 0.26066114$$

Thus

$$\delta_4 = 1.42237110$$

and

$$\delta_6 = 3.5719188$$

Step 6) Solve for X_1 and X_2 according to Rule 4 by replacing the δ's with their known values.

a) Write the equations for δ_4 and δ_6.

$$\delta_4 = X_1^{\ln 0.4}(\delta_3 + \delta_4)$$

$$\delta_6 = X_2^{\ln 0.6}(\delta_5 + \delta_6)$$

b) Substitute in the known values for the δ's.

$$1.4223711 = X_1^{-0.916290700}(1 + 1.4223711)$$

$$3.5719188 = X_2^{-0.05108256}(1 + 3.5719188)$$

c) Solve the above equations for X_1 and X_2.

$$X_1 = 60.913084$$

$$X_2 = 280.36967$$

Step 7) Use the above values to solve the objective function.

a) Find the optimal GP values of N_1 and N_2 by substituting in the values for the ln X_j.

$$N_1 = \ln X_1 = \ln 60.913084 = 4.109448$$

$$N_2 = \ln X_2 = \ln 280.36967 = 5.636109$$

b) Therefore, the optimal GP value of the objective function is

maximum reliability: R = 0.92195686

Phase 3: Integer Solution

Components cannot be split into fractions, therefore an integer solution is required. This phase will consist of four steps that will ultimately provide the maximum reliability integer component solution.

Step 1) Round each N_j down to the nearest integer, and write its value.

$$N_1 = 4$$

$$N_2 = 5$$

Step 2) This value will more than satisfy the original constraint; however, it may not be the optimal combination of variables that both satisfies the

constraint and has the maximum reliability. Determine the cost and reliability.

cost = \$5(4) + \$7(5) = \$55.00 < \$60.00

$$\text{reliability} = \left(1 - 0.4^4\right)\left(1 - 0.6^5\right) = 0.8986307$$

Step 3) Determine the number of combinations that must be compared to ensure that the optimal solution is found.

a) Subtract 1 from each integer value found in Step 1. Determine the new cost for each element.

cost of element 1 = \$5(3) = \$15.00

cost of element 2 = \$7(4) = \$28.00

b) Use the following formula developed by Steven A. Strauss to find the number of combinations to be checked to ensure that the maximum reliability is found subject to the maximum allowed cost.

$$\pm \text{ combination } N_j = \pm \text{ integer of } \left[\frac{(\text{budget limitation} - \text{costnew})}{(\text{cost of jth element})}\right] - \text{integer } N_j + 1$$

where

costnew = the cost of each element found in Step 3a, plus all the other element costs except for the jth element (the one whose optimal number of combinations is currently being calculated)

Therefore,

$\pm$ combination N_1 = $\pm$ integer of

$$\left[\frac{(\$60) - (\$28)}{\$5}\right] - 4 + 1 = \pm 3$$

$\pm$ combination N_2 = $\pm$ integer of

$$\left[\frac{(\$60) - (\$15)}{\$7}\right] - 5 + 1 = \pm 2$$

c) Find all combinations for each variable as determined in Step 3b. The combinations are also a sensitivity analysis for cost and reliability. Table 4.1 shows the combinations of N_1 and N_2 which result for example (1.2).

Step 4) Select the feasible combination with the maximum reliability. This will be the optimal integer solution.

maximum reliability: $R^* = 0.9127963$

$N_1^* = 5$

$N_2^* = 5$

Table 4.1 Maximum Integer Combinations

N_1	N_2	COST (IN DOLLARS)	RELIABILITY
2	3	31.00	0.6585600
2	4	38.00	0.7311360
2	5	45.00	0.7746816
2	6	52.00	0.8008090
2	7	59.00	0.8164854
3	3	36.00	0.7338240
3	4	43.00	0.8146944
3	5	50.00	0.8632166
3	6	57.00*	0.8923300
3	7	64.00	0.9097980
4	3	41.00	0.7639296
4	4	48.00	0.8481178
4	5	55.00*	0.8986307
4	6	62.00*	0.9289384
4	7	69.00	0.9471230
5	3	46.00	0.7759718
5	4	53.00	0.8614871
5	5	60.00*	0.9127963
5	6	67.00*	0.9435818
5	7	74.00	0.9620531
6	3	51.00	0.7639296
6	4	58.00	0.8668348
6	5	65.00*	0.9184625
6	6	72.00*	0.9494391
6	7	79.00	0.9680251

* denotes those combinations which are not feasible since they are more than the maximum cost limitation. The values for the combinations when N_1 is 1 and 7 are not included as the reliabilities are too low, or the cost exceeds the budget limitation.

Chapter 5

SOLUTIONS TO APPLIED SAMPLE RELIABILITY PROBLEMS

A generalized computer algorithm, as described in Chapters 3 and 4, was developed for a particular class of nonlinear reliabilty design problems and is presented in Appendix B. The computer algorithm is written in MICROSOFT Quickbasic for IBM-compatible microcomputers.

The algorithm is built around 5 areas: (1) substitution and manipulation of terms and variables to rewrite the original problem into standard GP form; (2) the use of GP to start solving for the δ values; (3) the use of the Brent-Dekker method to solve for the remaining unkown δ values; (4) the use of GP to solve for the optimal noninteger answer; and (5) solving for the optimal integer solution.

The type of reliability problems that this algorithm handles (minimization or maximization) will always remain at one DD, no matter how many variables are added. This is due to the substitution and manipulation that is done. Whenever another variable is added, one more constraint with two terms (one with an artificial variable) will also be added.

Currently the computer algorithm is limited to four variables. The results of running sample problems using the computer algorithm are addressed below.

Minimization Problems

Most industries, like the Department of Defense, are limited on how much they can spend to either purchase new systems or upgrade the present ones. Electronic equipment must therefore have a high degree of reliability with a reasonable cost in order to be competitive in today's world of advanced technology.

The following problems are indicative of the reliability constraints imposed on civilian contractors for electronic equipment used by the military. The first two problems were developed by COL Arbogast, a career Signal Officer currently stationed at the United States Military Academy. The third problem is one that was used as a sample problem in a published reliability text book (Dhillon, Balbir 1983). Three minimization problems are addressed below as test cases. There is a separate test problem depicting each number of variables solvable by the computer algorithm. For each test problem, the original problem will be stated, along with both the GP and integer solutions.

Two-Variable Problem

The first problem depicts a system that is comprised of two elements. Table 5.1 provides both the optimal GP and integer solutions. The original problem is shown below.

minimize cost: $C = \$40N_1 + \$60N_2$

subject to: $\left(1 - .02^{N_1}\right)\left(1 - .01^{N_2}\right) \geq .98$

Table 5.1 Minimum Two-Variable Solutions

Solution	*GP*	*Integer*
Reliability	0.979998	0.989604
Minimum Cost	$115.51	$149.00
N_1 Value	1.228815	2
N_2 Value	0.9617115	1

Three-Variable Problem

This problem depicts a system that is comprised of three elements. Table 5.2 depicts both the optimal GP and integer solutions for the problem. The original problem is shown below.

minimize cost: $C = \$40N_1 + \$52N_2 + \$60N_3$

subject to: $\left(1 - 0.01^{N_1}\right)\left(1 - 0.006^{N_2}\right)\left(1 - 0.003^{N_3}\right) \geq 0.97$

Table 5.2 Minimum Three-Variable Solutions

Solution	GP	Integer
Reliability	0.969840	0.990919
Minimum Cost	$133.84	$192.00
N_1 Value	1.020958	2
N_2 Value	0.888597	1
N_3 Value	0.779838	1

Four-Variable Problem

This problem taken from Reliability Engineering in Systems Design and Operation (pages 114-117), by Balbir Dhillon, depicts a system that is comprised of four elements. Table 5.3 depicts both the optimal GP and integer solutions for the problem. The answers derived by the computer program differ slightly from the ones given in the test book. The answers that are

different in the text book are in parenthesis by the computer answers. It appears that round off error may cause this difference. The original problem is shown below.

minimize cost: $C = \$5N_1 + \$10N_2 + \$8N_3 + \$2N_4$

subject to: $$\left(1 - 0.04^{N_1}\right)\left(1 - 0.03^{N_2}\right)\left(1 - 0.02^{N_3}\right)\left(1 - .05^{N_4}\right) \geq 0.98$$

Table 5.3 Minimum Four-Variable Solutions

Solution	GP	Integer
Reliability	0.980302	0.984330
Minimum Cost	$95.54	$100.00 ($98)
N_1 Value	5.611212	6
N_2 Value	3.924052	4
N_3 Value	3.859593	4
N_4 Value	8.332921	9 (8)

Maximization Problems

Cost budgets are an important part of the total system planning, especially when dealing with electronic pieces of

equipment. Two of the following problems were developed by COL Arbogast, while the third was taken from the article published by Federowicz and Mumzudar. Three maximization problems will be addressed as test cases; one for each number of variables solvable by the computer algorithm.

Two-Variable Problem

The first problem is one depicting a system that is comprised of two elements. Table 5.4 provides both the optimal GP and integer solutions. The original problem is shown below.

maximize reliability: $R = \left(1 - 0.02^{N_1}\right)\left(1 - 0.01^{N_2}\right)$

subject to: $\$40N_1 + \$60N_2 \leq \$200$

Table 5.4 Maximum Two-Variable Solutions

Solution	GP	Integer
Reliability	0.999635	0.999500
Minimum Cost	$200.00	$200.00
N_1 Value	2.233251	2
N_2 Value	1.844499	2

Three-Variable Problem

This problem depicts a system that is comprised of three elements. Table 5.5 depicts both the optimal GP and integer solutions for the problem. The original problem is shown below.

maximize reliability: $R = \left(1 - 0.01^{N_1}\right)\left(1 - 0.004^{N_2}\right)\left(1 - 0.003^{N_3}\right)$

subject to: $\$40N_1 + \$52N_2 + \$60N_3 \leq \$250$

Table 5.5 Maximum Three-Variable Solutions

Solution	GP	Integer
Reliability	0.999546	0.9968843
Minimum Cost	$250.00	$244.00
N_1 Value	1.928819	2
N_2 Value	1.59408	2
N_3 Value	1.49925	1

Four-Variable Problem

This problem taken from the article published by Federowicz and Mazumdar, depicts a system that is comprised of four elements. Table 5.3 depicts both the optimal GP and the integer solutions for the problem. The answers derived by the comput r program differ slightly from the ones given in the test article. Roundoff error and the way combinations are checked appear to be the reason for the computer not attaining the correct optimal answer. The answers that differ are noted with the article answer in parentheses. The original problem is shown below.

maximize reliability: $R = \left(1 - 0.20^{N_1}\right)\left(1 - 0.30^{N_2}\right)$

$$\left(1 - 0.25^{N_3}\right)\left(1 - 0.15^{N_4}\right)$$

subject to: $\$1.20N_1 + \$2.3N_2$

$$+ \$3.40N_3 + \$4.50N_4 \leq \$47$$

Table 5.6 Maximum Four-Variable Solutions

Solution	GP	Integer
Reliability	0.993107	0.988735 (0.99)
Minimum Cost	$47.00	$43.40 ($45.80)
N_1 Value	4.522890	4 (6)
N_2 Value	5.265519	5
N_3 Value	4.393139	4
N_4 Value	3.227793	3

Chapter 6

CONCLUSIONS

Technology is an ever changing area of science. As equipment becomes more sophisticated, the need for reliable and cost effective elements increases. A reliability model that is quick, accurate, and efficient is an important part of resource management. It is evident that the generalized computer algorithm can be a useful microeconomic tool in terms of convenience, speed of results, and reduction of computational error in calculating the optimal combinations of system elements.

The strength of this computer algorithm is that it provides a method for solving difficult problems that are currently either guessed at, or require an excessive amount of time to solve either by hand or computer. The program is easy to use and solves each problem very quickly. The algorithm has a slight weakness. When the costs for each element are relatively small, with little difference, and the failure rates are also fairly close, the computer sometimes does not recognize a difference between adding or subtracting one element. This appears to happen occassionally when there are four variables.

One area not fully explored was that of sensitivity analysis. The computer algorithm presently checks different combinations of the variables, but only with their present cost and failure rate values. The algorithm should be expanded to include a method that would determine how sensitive each element is to its own cost and failure, as well as that of the total system.

Another important area for further research is that of expanding the computer algorithm to include more than four variables. This means developing a method to determine the appropriate function to be solved when searching for δ_2. The subroutine that searches for the unknown δ values should also be refined. The Brent-Dekker method is very sensitive when converging on the positive root. When the method finds a positive root, the convergence factor becomes less concise as the number of variables increases. It is also noted that as the required minimum reliabilities become more precise, the less concise the criteria for the convergence factor can be. A lower convergence factor is therefore necessary, otherwise, the computer tends to oscilliate around the positive root.

The program could also be modified to solve problems with additional constraints. For instance, there may be a

size capacity which will require a constraint limiting the total number of components.

Another area for further research is that of improving the manipulation of the original problem to determine whether a different method will reduce the problem to a zero DD, GP problem. If the problem could be reduced to zero DD it would cut the solving time in half as the Brent-Dekker method could then be eliminated from the program.

The concept of using manipulation and substitution to form messy problems into ones solvable by GP, and then using a nonlinear method such as Brent-Dekker to solve the inherent one DD mathematical problem significantly broadens the range of possible applications of GP. This in itself is an important area for further research.

The main contribution of this thesis is that it will provide a simple-to-use, generalized computer algorithm for solving a large class of nonlinear equations. In addition, the algorithm guarantees that the GP solution will be globally optimal. The integer solution cannot guarantee global optimality, however, in most cases it will provide the optimum local integer combination.

REFERENCES

Allen, Patrick D. and David W. Baker. (N.d.) "An Introduction to Geometric Programming." U.S. Army Concepts Analysis Agency, Bethesda, Md.

Abel-Hameed, Mohamed S., Erhan Cinlar, and Joseph Quinn, Eds. 1984. Reliability Theory and Models. New York: Academic Press.

Avriel, Mordecai, Ed. 1980. Advances in Geometric Programming. New York: Plenuirn Press.

Dhillon, Babbin S. 1983. Power System Reliability, Safety and Management. Ann Arbor, Mich.: The Butterworth Group.

Dhillon, Balbir S. 1983. Reliability Engineering in Systems Design and Operation. New York: Van Nostrand Reinhold.

Duffin, R. J., E. L. Peterson, and C. Zener. 1967. Geometric Programming. New York: Wiley.

Federowicz, A. J. and M. Mazumdar. 1968. "Use of Geometric Programming to Maximize Reliability Achieved by Redundancy." Operations Research 15: 948-54.

Hillier, Frederick S. and Gerald J. Lieberman. 1986. "Reliability." Ch. 21 in Introduction to Operations Research. 4th Ed. Oakland, Calif.: Holden-Day.

Shooman, Martin L. 1968. Probabilistic Reliability: An Engineering Approach. New York: McGraw-Hill

Tersine, Richard J. 1980. "Product Decisions". Ch. 7 Production/Operations Management: Concepts, Structure, and Analysis. New York: North Holland.

Woolsey, R. E. D. and Huntington S. Swanson. 1969. Operations Research for Immediate Application: A Quick and Dirty Manual. New York: Harper and Row.

Appendix A

FOUR RULES FOR GEOMETRIC PROGRAMMING

Rule 1: The form of the optimal solution of any posynomial GP problem is:

Value of the Objective Function* =

$$\left(\text{coef. of first term in obj. function}/\delta_1\right)^{\delta_1} \times \ldots \times \left(\text{coef. of last term in obj. function}/\delta_{last}\right)^{\delta_{last}} \times$$

$$\left(\text{coef. of first term in constraint}/\delta_{last+1}\right)^{\delta_{last+1}} \times \ldots \times \left(\text{coef. of last term in constraint}/\delta_{last+m}\right)^{\delta_{last+m}} \times$$

$$(\text{sum of } \delta\text{'s in constraint})^{(\text{sum of } \delta\text{'s in constraint})}$$

Rule 2: The exponent matrix is constructed in the following way:

Rule 2A: The sum of contributions to cost in the objective function = 1.

$$\delta_1 + \delta_2 + \ldots + \delta_{last} = 1$$

Rule 2B: For each primal variable the equations in the exponent matrix are:

(power of variable j in term 1) X δ_1 +

(power of variable j in term 2) X δ_2 +

...

(power of variable j in last term) X $\delta_{last+m} = 0$

Rule 3: At optimality

Value of the Objective Function* =

(first term in obj. function/δ_1^*) =

(second term in obj. function/δ_2^*) =

...

(last term in obj. function/δ_{last}^*)

Rule 4: At optimality, for each constraint

δ_j = (jth term in constraint) X

(sum of δ's in constraint)

Appendix B

COMPUTER ALGORITHM FOR SOLVING RELIABILITY PROBLEMS WITH GP USING MICROSOFT QUICKBASIC

The reliability program developed for this thesis is designed to run on an IBM-compatible personal computer using a software package called Microsoft Quickbasic. To use the Microsoft Quickbasic package, the user must have access to a computer with at least 384k memory. The computer program could easily be modified and written in the Basic language of the user's personal computer if Microsoft Quickbasic is not available.

To run the program, the program is first read into the computer and an executable file is made. The program can be then be executed by typing the file name (i.e., RELY) at the computer A prompt.

The program is designed to be user interactive. Once the program is loaded, the computer will begin by asking the user for input. Appendix C depicts a sample computer run.

The reliability computer algorithm using Microsoft Quickbasic is listed below.

```
'*******************************************************
'   WELCOME TO CPT OATNEY'S RELIABILITY SOLVING      *
'   PROGRAM.  THIS PROGRAM WILL EITHER MINIMIZE      *
'   COST SUBJECT TO A REQUIRED RELIABILITY OR        *
'   MAXIMIZE RELIABILITY SUBJECT TO A BUDGET         *
'   LIMITATION USING GP.  THE PROGRAM WILL HANDLE    *
'   FROM 2 TO 4 VARIABLES.  THE FINAL ANSWER, IN     *
'   INTEGER FORM, WILL INCLUDE:  THE REQUIRED        *
'   QUANTITY FOR EACH VARIABLE AND EITHER THE        *
'   MINIMUM COST OR MAXIMUM RELIABILITY OBTAINED     *
'   FOR THE DEFINED PROBLEM.                         *
'*******************************************************



OPTION BASE 1
DEFINT I-L,N
DEFDBL A-D,F,M,P
DIM COF(10),PROB(10),XX(10),YY(10),DELTA(20)
DIM NO(10),PROBNEW(10),DIFFER(10),JCO(10)
DIM POWER(10),COSTNEW(10)



1       CLS
        GOSUB STARTINP


        IF YN$="N" THEN GOSUB MAXIM ELSE GOSUB MINIM


        GOSUB ROUNDOFF
        GOSUB VERIFY
        GOSUB PRINTOUT
        GOTO 9999
```

```
STARTINP:

        '****** THIS ROUTINE INITIALIZES THE PROGRAM *******
        '************ BY ASKING THE USER TO INPUT **********
        '************* THE ORIGINAL PROBLEM DATA ***********


10      INPUT; "IS THIS PROBLEM A MINIMIZATION? (Y/N) ",YN$
        IF YN$<>"Y" AND YN$<>"N" THEN GOTO 10
        PRINT ""

15      INPUT; "HOW MANY VARIABLES ARE THERE? (2-4) ",NVAR
        PRINT ""

16      PRINT "YOU HAVE INPUT ",NVAR
        PRINT "AS THE NUMBER OF VARIABLES"
        INPUT; "IS THIS VALUE ACCEPTABLE? (Y/N) ",YYNN$
        PRINT ""
        IF YYNN$="N" THEN GOTO 15
        IF YYNN$<>"Y" THEN GOTO 16

        FOR J=1 TO NVAR

20          CLS
            LOCATE 5,12
            PRINT "YOU ARE INPUTING VARIABLE #",J
            INPUT; "WHAT IS THE COST COEFFICIENT? ",COF(J)
            PRINT ""
            INPUT; "WHAT IS THE ASSOCIATED PROBABILITY OF
                    FAILURE? ",PROB(J)
            PRINT ""
            PRINT "YOU HAVE INPUT ";COF(J);" AND ";PROB(J)
            PRINT "AS THE COST AND FAILURE PROBABILITY"

30          LOCATE 15,12
            INPUT; "ARE THESE VALUES ACCEPTABLE? (Y/N) ",YNN$
            PRINT ""
            IF YNN$="N" THEN GOTO 20
            IF YNN$<>"Y" THEN GOTO 30
        NEXT J
        CLS

40      IF YN$="Y" THEN
            INPUT; "WHAT IS THE REQUIRED RELIABILITY? ",RELY
            PRINT ""
            PRINT "YOU HAVE INPUT ";RELY
            PRINT "AS THE REQUIRED RELIABILITY"
```

```
50          LOCATE 15,12
            INPUT; "IS THIS VALUE ACCEPTABLE? (Y/N) ',YNN$
            PRINT ""
            IF YNN$="N" THEN GOTO 40
            IF YNN$<>"Y" THEN GOTO 50
        ELSE

60          INPUT; "WHAT IS THE BUDGET LIMITATION? ",BUDG
            PRINT ""
            PRINT "YOU HAVE INPUT ";BUDG
            PRINT "AS THE BUDGET LIMITATION"

70          LOCATE 15,12
            INPUT; "IS THIS VALUE ACCEPTABLE? (Y/N) ",YNN$
            IF YNN$="N" THEN GOTO 60
            IF YNN$<>"Y" THEN GOTO 70
        END IF
        RETURN


MAXIM:

        '********** THIS SUBROUTINE PROVIDES THE **********
        '************ EQUATIONS NEEDED TO SOLVE ************
        '**************** FOR THE δ VALUES IN **************
        '**************** MAXIMIZATION PROBLEMS *************


        DELTA(1)=1
        FOR J = 1 TO NVAR
            COFF(J)=COF(J)/BUDG
            POWER(J) = -COFF(J)/LOG(PROB(J))
            DELTA(2*J+1)=1
        NEXT J
        GOSUB DFUN
        GOSUB SECSOLVE
        GOSUB XMAXSOLVE
        RETURN
```

```
MINIM:

        '********** THIS SUBROUTINE PROVIDES THE **********
        '************ EQUATIONS NEEDED TO SOLVE ************
        '**************** FOR THE δ VALUES IN **************
        '**************** MINIMIZATION PROBLEMS ************
        '************ IT ALSO SOLVES TWO-VARIABLE **********
        '******* PROBLEMS USING THE QUADRATIC EQUATION *****


        CON = -1
        DELTA(1) = 1
        AA = (1-RELY)
        BB = 0 : CC = 0 : DD = 0
        FOR J = 1 TO NVAR
           DELTA(2*J+1) = -COF(J)/LOG(PROB(J))
           BB = BB-DELTA(2*J+1)*RELY
           CON = CON*DELTA(2*J+1)*RELY
           FOR IJ = J+1 TO NVAR
              IF IJ>NVAR THEN GOTO 100
              CC = CC-DELTA(2*J+1)*DELTA(2*IJ+1)*RELY
              FOR IJK = IJ+1 TO NVAR
                 IF IJK>NVAR THEN GOTO 100
                 DD = DD-DELTA(2*J+1)*DELTA(2*IJ+1)*
                      DELTA(2*IJK+1)*RELY
              NEXT IJK
           NEXT IJ

100     NEXT J
        IF NVAR=2 THEN
           D2 = (-BB+SQR(BB^2-4*AA*CON))/(2*AA)
           GOSUB XMINSOLVE
           RETURN
        END IF
        IF NVAR>2 THEN GOSUB DFUN
        GOSUB SECSOLVE
        GOSUB XMINSOLVE
        RETURN
```

```
DFUN:

        '******* THIS SUBROUTINE DEFINES THE FUNCTION ******
        '********* TO BE USED WHEN SOLVING FOR THE *********
        '************ REAL POSITIVE ROOT OF THE ************
        '************ UNKNOWN S VALUES IN EITHER ***********
        '******* MAXIMIZATION OR MINIMIZATION PROBLEMS *****



        DEF FNFUNC (D2)
        IF YN$="Y" THEN
           FNFUNC = AA*D2^NVAR+BB*D2^(NVAR-1)+CC*D2^(NVAR-2)
                    +DD*D2^(NVAR-3)*(NVAR-3)+CON
        ELSE
           DUM=1
           FOR J = 1 TO NVAR
              DUM = DUM*(((1+POWER(J)*D2)/
                    (POWER(J)*D2))^POWER(J))
           NEXT J
           FNFUNC = -EXP(1)+DUM
        END IF
        END DEF
        RETURN


SECSOLVE:

        '******** THIS SUBROUTINE FINDS THE POSITIVE *******
        '*********** REAL ROOT FOR THE UNKNOWN S'S *********
        '******* THE SUBROUTINE BEGINS BY REQUESTING *******
        '************ AN INITIAL STARTING BRACKET **********



220     PRINT "PLEASE PROVIDE A STARTING BRACKET [B,C]
                SEPARATED BY A COMMA "
        INPUT "WHICH YOU BELIEVE CONTAINS A ROOT ";B,C
        IF B<0 OR C<0 OR (B=0 AND C=0) THEN GOTO 220


240     PRINT "THANK YOU, YOUR STARTING BRACKET IS = ",B,C
        D=C:GOSUB 900
```

```
        '*************** MAIN PROGRAM STARTS HERE ***********
        '*** QUICK CHECK: IS FA AND FB OF OPPOSITE SIGNS? **

        A=C
        GOSUB 760

330     GOSUB 690
        IF FA*FB > 0 THEN GOTO 1350
        CONVERG=10^(-8+(2*(NVAR-2)))
        IF ABS(FB*FD)<CONVERG THEN
           IF ABS(FD)<ABS(FB) THEN
              B=D
              FB=FD
           END IF
        END IF

350     IF ABS(FB)<CONVERG THEN 1200
        GOSUB 830
        IF ABS(FC)=>ABS(FB) THEN GOTO 480
        A=B
        GOSUB 760
        B=C
        GOSUB 690
        C=A
        GOSUB 830
        GOTO 350


        '********** PRINT CURRENT VALUES OF B,A,FB,FA *******

480     PRINT
        D=(FB*A-FA*B)/(FB-FA)
        PRINT
        GOSUB 900
        A=B
        PRINT
        GOSUB 760
        PRINT
        M=(B+C)/2
        GOSUB 920
        GOTO 970


        '****************** CALCULATE FB ******************

690     FB=FNFUNC(B)
        PRINT
        RETURN
```

```
        '******************** CALCULATE FA ******************

760     FA=FNFUNC(A)
        PRINT
        RETURN

        '******************** CALCULATE FC ******************

830     FC=FNFUNC(C)
        PRINT
        RETURN

        '******************** CALCULATE FD ******************

900     FD=FNFUNC(D)
        PRINT
        RETURN

        '******************** CALCULATE FM ******************

920     FM=FNFUNC(M)
        PRINT
        RETURN

        '******* PRINT CURRENT VALUES OF D,B,M,AND C *******

970     PRINT
        PRINT "THE CURRENT VALUES OF D,B,M,C ARE = ",D,B,M,C
        FXX=FD*FB
        FYY=FD*FM
        FZZ=FC*FD
        IF FYY<=0 THEN
           IF ABS(FD)<ABS(FM) THEN
              B=D:FB=FD
              C=M:FC=FM
           ELSE
              C=D:FC=FD
              B=M:FB=FM
           END IF
           A=C:FA=FC
           GOTO 330
```

```
        ELSE
           IF FZZ<=0 THEN
              B=C:C=A:A=B
              FB=FC:FC=FA:FA=FB
              GOTO 1050
           END IF
        END IF
        PRINT

1050    IF (D>B AND D<M) AND FXX<0 THEN GOTO 1060
        IF (D<B AND D>M) AND FXX<0 THEN GOTO 1060
        GOTO 1100

1060    B=D
        FB=FD
        GOTO 1120

1100    B=M
        FB=FM

1120    IF (FB*FC)=>0 THEN C=A:GOSUB 830
        GOTO 330


        '****************** PRINT RESULTS ******************

1200    CLS
        PRINT
        PRINT "THE OPTIMAL VALUE OF THE UNKNOWN X IS = ",B
        PRINT
        D2=B
        GOTO 1400


        '************ TELL USER BRACKET IS NO GOOD **********

1350    PRINT
        PRINT "ROOT NOT BRACKETED OR SECANT METHOD WILL NOT
               FIND ROOT, TRY AGAIN"
        GOTO 220


        '******** USER IS FINISHED WITH SUBROUTINE *********

1400    RETURN
```

```
XMAXSOLVE:

        '****** THIS SUBROUTINE SOLVES FOR THE X VALUES ****
        '************ IN MAXIMIZATION PROBLEMS *************



        FOR J = 1 TO NVAR
           DELTA(2*J+2)=D2*POWER(J)
           XX(J) = (LOG(DELTA(2*J+2)/(DELTA(2*J+1)
                   +DELTA(2*J+2))))/LOG(PROB(J))
        NEXT J
        RETURN


XMINSOLVE:

        '****** THIS SUBROUTINE SOLVES FOR THE X VALUES ****
        '************ IN MINIMIZATION PROBLEMS *************



        FOR J = 1 TO NVAR
           DELTA(2*J)=D2
           XX(J) = (LOG(DELTA(2*J+1)/(DELTA(2*J)
                   +DELTA(2*J+1))))/LOG(PROB(J))
        NEXT J
        RETURN


ROUNDOFF:

        '********** THIS SUBROUTINE ROUNDS THE GP **********
        '************ SOLUTION TO INTEGER VALUES ***********



        FOR J = 1 TO NVAR
           YY(J)=INT(XX(J))
           IF INT(XX(J))<XX(J) AND YN$="Y" THEN
              YY(J)=INT(XX(J))+1
        NEXT J
        RETURN
```

```
VERIFY:

        '******** THIS SUBROUTINE FINDS AND VERIFIES *******
        '*********** THE OPTIMAL INTEGER SOLUTION  *********


        PROBOPT=1:COSTOPT=0
        FOR J = 1 TO NVAR
           COSTOPT=COSTOPT+COF(J)*YY(J)
           PROBOPT=PROBOPT*(1-PROB(J)^YY(J))
        NEXT J

        IF YN$="N" THEN
           FOR J = 1 TO NVAR
              NO(J)=YY(J)-1
           NEXT J
           FOR J = 1 TO NVAR
              COSTNEW(J)=0
              FOR K = 1 TO NVAR
                 IF J<>K THEN
                    COSTNEW(J)=COSTNEW(J)+(COF(J)*NO(K))
              NEXT K
              DIFFER(J) = INT((BUDG-COSTNEW(J))/
                          COF(J))-YY(J)+1
           NEXT J
        END IF

        IF YN$="Y" THEN
           FOR J = 1 TO NVAR
              NO(J)=YY(J)+1
           NEXT J
           FOR J = 1 TO NVAR
              PROBNEW(J)=1
              FOR K = 1 TO NVAR
                 IF J<>K THEN PROBNEW(J)=PROBNEW(J)*
                                         (1-PROB(K)^NO(K))
              NEXT K
              DIFFER(J)=YY(J)-INT((LOG(1-(RELY/
                        PROBNEW(J))))/(LOG(PROB(J))))
        NEXT J
        END IF
```

```
        '******* TEST FOR CONSTRAINT VIOLATION AND *********
        '********** A BETTER OBJECTIVE FUNCTION ************


          FOR K = 1 TO NVAR: NO(K)=YY(K): NEXT K


        '******* DETERMINE THE NUMBER OF COMBINATIONS ******
        '************* TO BE EXAMINED TO FIND **************
        '*************** THE OPTIMAL SOLUTION ***************



         FOR J1 = YY(1)-DIFFER(1) TO YY(1)+DIFFER(1)
            JCO(1)=J1
            IF J1<1 THEN GOTO 9900
            FOR J2 = YY(2)-DIFFER(2) TO YY(2)+DIFFER(2)
               JCO(2)=J2
               IF J2<1 THEN GOTO 9800
               IF NVAR=2 THEN GOSUB COMPARE: GOTO 9800
               FOR J3 = YY(3)-DIFFER(3) TO YY(3)+DIFFER(3)
                  JCO(3)=J3
                  IF J3<1 THEN GOTO 9700
                  IF NVAR=3 THEN GOSUB COMPARE: GOTO 9700
                  FOR J4=YY(4)-DIFFER(4) TO YY(4)+DIFFER(4)
                     JCO(4)=J4
                     IF J4<1 THEN GOTO 9600
                     GOSUB COMPARE
9600              NEXT J4
9700           NEXT J3
9800        NEXT J2
9900     NEXT J1
        RETURN


COMPARE:

        '************ THIS SUBROUTINE FINDS THE ************
        '*************** OPTIMAL COMBINATION ***************



        IF YN$="N" THEN
           COSTCALC=0
           FOR JJ = 1 TO NVAR
              COSTCALC=COSTCALC+COF(JJ)*JCO(JJ)
           NEXT JJ
```

```
            IF COSTCALC>BUDG THEN RETURN
            PROBCALC=1
            FOR JJ = 1 TO NVAR
               PROBCALC=PROBCALC*(1-PROB(JJ)^JCO(JJ))
            NEXT JJ
            IF PROBCALC<PROBOPT THEN RETURN
               PROBOPT = PROBCALC
            FOR JJ = 1 TO NVAR
               NO(JJ)=JCO(JJ)
            NEXT JJ
         ELSE
            PROBCALC=1
            FOR JJ = 1 TO NVAR
               PROBCALC=PROBCALC*(1-PROB(JJ)^JCO(JJ))
            NEXT JJ
            IF PROBCALC<RELY THEN RETURN
            COSTCALC=0
            FOR JJ = 1 TO NVAR
               COSTCALC=COSTCALC+COF(JJ)*JCO(JJ)
            NEXT JJ
            IF COSTCALC<COSTOPT THEN
               COSTOPT = COSTCALC
               FOR JJ = 1 TO NVAR
                  NO(JJ)=JCO(JJ)
               NEXT JJ
            END IF
         END IF
         RETURN

PRINTOUT:

         '********** THIS SUBROUTINE TELLS THE USER *********
         '*********** WHAT THE OPTIMAL SOLUTION IS **********


         PRINT "THE ORIGINAL PROBLEM WAS:"
         IF YN$="N" THEN
            PRINT "MAXIMIZE RELIABILITY";
            FOR JJ = 1 TO NVAR-1
               PRINT "(1-(";PROB(JJ);"^N(";JJ"))*";
            NEXT JJ
            PRINT "(1-(";PROB(NVAR);"^N(";NVAR"))";
            PRINT " S.T. ";
            FOR JJ = 1 TO NVAR-1
               PRINT COF(JJ);"*N(";JJ;") +";
```

```
      NEXT JJ
      PRINT COF(NVAR);"*N(";NVAR;") <= ";BUDG
      PRINT
   ELSE
      PRINT "MINIMIZE COST";
      FOR JJ = 1 TO NVAR-1
         PRINT COF(JJ);"N(";JJ;") +";
      NEXT JJ
      PRINT COF(NVAR);"*N(";NVAR;")"
      PRINT " S.T. ";
      FOR JJ = 1 TO NVAR-1
         PRINT "(1-(";PROB(JJ);"^N(";JJ"))*";
      NEXT JJ
      PRINT "(1-(";PROB(NVAR);"^N(";NVAR")) => ";RELY
      PRINT
   ENDIF
   COSTOLD=0 : COSTCALOPT=0 : PROBOLD=1 : PROBCALOPT=1
   PRINT "THE GP SOLUTION FOR THE PROBLEM HAS VALUES
         OF"
   FOR J = 1 TO NVAR
      PRINT "  N(";J;")=";XX(J)
      COSTOLD=COSTOLD+XX(J)*COF(J)
      COSTCALOPT=COSTCALOPT+NO(J)*COF(J)
      PROBOLD=PROBOLD*(1-PROB(J)^XX(J))
      PROBCALOPT=PROBCALOPT*(1-PROB(J)^NO(J))
   NEXT J
   PRINT "  WITH AN OPTIMAL COST OF ";COSTOLD
   PRINT "  AND A RELIABILITY OF ";PROBOLD
   PRINT
   PRINT "THE OPTIMAL INTEGER SOLUTION HAS VALUES OF "
   FOR J = 1 TO NVAR
      PRINT "  N(";J;")=";NO(J)
   NEXT J
   PRINT "  WITH AN OPTIMAL COST OF ";COSTCALOPT
   PRINT " AND A RELIABILITY OF ";PROBCALOPT

9997  INPUT "DO YOU WANT A HARD COPY OF THIS RESULT?
         (Y/N) ",NY$
   IF NY$="N" THEN GOTO 9998
   IF NY$<>"Y" THEN GOTO 9997

   LPRINT
   LPRINT "THE OPTIMAL VALUE OF $_2$ IS = ",B
   LPRINT
   LPRINT "THE ORIGINAL PROBLEM WAS:"
   IF YN$="N" THEN
      LPRINT "MAXIMIZE RELIABILITY";
      FOR JJ = 1 TO NVAR-1
```

```
            PRINT "(1-(";PROB(JJ);"^N(";JJ"))*";
         NEXT JJ
         LPRINT "(1-(";PROB(NVAR);"^N(";NVAR"))";
         LPRINT " S.T. ";
         FOR JJ = 1 TO NVAR-1
            LPRINT COF(JJ);"*N(";JJ;") +";
         NEXT JJ
         LPRINT COF(NVAR);"*N(";NVAR;") <= ";BUDG
         LPRINT
      ELSE
         LPRINT "MINIMIZE COST";
         FOR JJ = 1 TO NVAR-1
            LPRINT COF(JJ);"*N(";JJ;") +";
         NEXT JJ
         LPRINT COF(NVAR);"*N(";NVAR;")"
         LPRINT " S.T. ";
         FOR JJ = 1 TO NVAR-1
            LPRINT "(1-(";PROB(JJ);"^N(";JJ"))*";
         NEXT JJ
         LPRINT "(1-(";PROB(NVAR);"^N(";NVAR")) => ";RELY
         LPRINT
      END IF
      LPRINT "THE GP SOLUTION FOR THE PROBLEM HAS VALUES
              OF"
      FOR J = 1 TO NVAR
         LPRINT "  N(";J;")=";XX(J)
      NEXT J
      LPRINT "  WITH AN OPTIMAL COST OF ";COSTOLD
      LPRINT "  AND A RELIABILITY OF ";PROBOLD
      LPRINT
      LPRINT "THE OPTIMAL INTEGER SOLUTION HAS VALUES
              OF "
      FOR J = 1 TO NVAR
         LPRINT "  N(";J;")=";NO(J)
      NEXT J
      LPRINT "  WITH AN OPTIMAL COST OF ";COSTCALOPT
      LPRINT "  AND A RELIABILITY OF ";PROBCALOPT

9998  RETURN

9999  INPUT; "DO YOU WISH TO DO ANOTHER PROBLEM? (Y/N) ",
              NY$
      IF NY$="Y" THEN GOTO 1
      IF NY$<>"N" THEN GOTO 9999

      END
```

Appendix C

SAMPLE COMPUTER RUN

Computer Questions	Input
IS THIS PROBLEM A MINIMIZATION? (Y/N)	Y
HOW MANY VARIABLES ARE THERE? (2-4)	3
YOU HAVE INPUT 3 AS THE NUMBER OF VARIABLES	
IS THIS VALUE ACCEPTABLE? (Y/N)	Y
YOU ARE INPUTTING VARIABLE 1	
WHAT IS THE COST COEFFICIENT?	5
WHAT IS THE ASSSOCIATED PROBABILITY OF FAILURE?	.4
YOU HAVE INPUT 5 AND .4	
AS THE COST AND FAILURE PROBABILITY	
ARE THESE VALUES ACCEPTABLE? (Y/N)	Y
YOU ARE INPUTTING VARIABLE 2	
WHAT IS THE COST COEFFICIENT?	7
WHAT IS THE ASSSOCIATED PROBABILITY OF FAILURE?	.6
YOU HAVE INPUT 7 AND .6	
AS THE COST AND FAILURE PROBABILITY	
ARE THESE VALUES ACCEPTABLE? (Y/N)	Y
YOU ARE INPUTTING VARIABLE 3	
WHAT IS THE COST COEFFICIENT?	6
WHAT IS THE ASSSOCIATED PROBABILITY OF FAILURE?	.5
YOU HAVE INPUT 6 AND .5	
AS THE COST AND FAILURE PROBABILITY	
ARE THESE VALUES ACCEPTABLE? (Y/N)	Y
WHAT IS THE REQUIRED RELIABILITY?	.9
YOU HAVE INPUT .90	
AS THE REQUIRED RELIABILITY	
IS THIS VALUE ACCEPTABLE? (Y/N)	Y
PLEASE PROVIDE A STARTING BRACKET	
[B,C] SEPARATED BY A COMMA	
WHICH YOU BELIEVE CONTAINS A ROOT	
THANK YOU, YOUR STARTING BRACKET IS =	0,1000

Computer Output

THE OPTIMAL VALUE OF δ_2 IS = 250.421517

THE ORIGINAL PROBLEM WAS
MINIMIZE COST 5 * N(1) + 7 * N(2) + 6 * N(3)
S.T. (1 - (.4^N(1))*(1 - (.6^N(2))*(1 - (.5^N(3)) => .90

THE GP SOLUTION FOR THE PROBLEM HAS VALUES OF
N(1) = 4.199369
N(2) = 5.792162
N(3) = 4.903512

WITH AN OPTIMAL COST OF 90.96304798
AND A RELIABILITY OF .91409482752

THE OPTIMAL INTEGER SOLUTION HAS VALUES OF
N(1) = 5
N(2) = 6
N(3) = 5

WITH AN OPTIMAL COST OF 97
AND A RELIABILITY OF .896896312

END

5-87

DTIC